The Plant Propagation Algorithm on Timetables: First Results

Geleijn, R.; van der Meer, M.; van der Post, Q.; van den Berg, D.

Publication date
2019

Document Version
Author accepted manuscript

Link to publication

Citation for published version (APA):
Geleijn, R., van der Meer, M., van der Post, Q., & van den Berg, D. (2019). *The Plant Propagation Algorithm on Timetables: First Results*. 2-3. Abstract from EvoStar 2019, Leipzig, Saxony, Germany. https://arxiv.org/abs/1907.12698

The Plant Propagation Algorithm on Timetables: First Results

Romi Geleijn*
Institute of Interdisciplinary Studies
University of Amsterdam, The Netherlands
romi.geleijn@student.uva.nl

Marrit van der Meer*
Institute of Interdisciplinary Studies
University of Amsterdam, The Netherlands
marrit.vandermeer@student.uva.nl

Quinten van der Post
Minor Programmeren
University of Amsterdam, The Netherlands
q.f.vanderpost@uva.nl

Daan van den Berg
Minor Programmeren, Docentengroep IvI
University of Amsterdam, The Netherlands
d.vandenberg@uva.nl

ABSTRACT

One Stochastic HillClimber and two implementations of the Plant Propagation Algorithm (PPA-1 and PPA-2) are applied to an instance of the University Course Timetabling Problem from the University of Amsterdam. After completing 10 runs of 200,000 objective function evaluations each, results show that PPA-1 outperforms the HillClimber, but PPA-2 makes the best timetables.

KEYWORDS

University Timetabling, UCTP, Evolutionary Algorithm, Plant Propagation Algorithm

1 INTRODUCTION

Universities all over the world are faced with the University Course Timetabling Problem (UCTP), an NP-hard constrained optimization problem, which means that an optimal solution for a realistically sized timetable cannot be found within any reasonable amount of time [1]. For these kinds of problems, exact algorithms are practically useless, but sufficiently good solutions can be produced by heuristic optimization algorithms such as Genetic Algorithms, Ant Colony Optimization and Tabu Search [1] [2] [3] [4] [5] [6].

In this preliminary investigation, performance of the Plant Propagation Algorithm (PPA), a bio-inspired meta-heuristic, is assessed when applied to the UCTP at the University of Amsterdam (UvA). Previous studies have applied PPA to the Uncapacitated Exam Scheduling Problem (UESP), which is related to UCTP as both are a subcategory of Academic Scheduling Problems, but also to the Traveling Salesperson Problem (TSP), another NP-hard constrained optimization problem [7][8]. The algorithm has not been subjected to the UCTP itself, but has performed well on a diverse array of combinatorial optimization problems and might be a promising candidate, given that we develop a suitable adaptation from its previous implementations [9] [10]. This study uses a(n anonymized) dataset from the UvA and evaluates the performance of three algorithms for optimally scheduling its courses. First, a simple Stochastic HillClimber (HC) algorithm (a.k.a. "stochastic local search") is implemented. Second, PPA-1 is a direct adaptation from its TSP-cousin to the timetabling problem [8]. PPA-2 finally, is an adaptation that stems from the seminal implementation of PPA on continuous functions [9], therefrom inheriting a somewhat smoother procreation strategy. The results of all three algorithms on this single real-world instance of UCTP are quantitatively compared.

*Both authors contributed equally to this work.

2 TIMETABLES AND OBJECTIVE VALUES

For this explorative study, data from 29 existing courses and 609 fictional UvA-students is used. Every student is enrolled in 1 to 5 courses, and no interdependencies between courses are currently implemented. Courses consist of zero or more plenary lectures, study groups and lab practicals ('labs'), the latter two activities occasionally being split up in equally sized *sessions* to meet capacity constraints. For instance, if there are 78 students enrolled in "Machine Learning II", but labs in this course can only accommodate 20 students at a time, four sessions of that single lab are scheduled to accommodate all enrolled students. All 129 course activities are scheduled in 7 rooms with varying capacities from the UvA's Science Park location, each having 4 time slots on all 5 weekdays, amounting to 140 weekly *room slots*. Neglecting symmetries and equivalences, these numbers of course activities and room slots give rise to $\frac{129!}{11!} \approx 3.4 * 10^{233}$ different timetable configurations just for one week, even for this reduced problem instance.

An initial timetable is created by assigning each course activity to exactly one randomly chosen room slot. Then, every student attending a course is assigned to all activities within the course; if a course activity is split up, one available session is selected at random. Once completed, this constraint-satisfying (or *'valid'*) initial timetable is assigned a base objective value of 1,000, after which three objective modifiers are applied. First, for every activity, any student number that exceeds the room size reduces the objective value by one. Second, for each student that has more than one course activity at any given time slot, one point is deducted for every excess activity. Third, if a course has its activities spread optimally over the week (e.g. two activities either on Monday-Thursday or Tuesday-Friday), 20 points are added to the timetable's objective value. Conversely, if the number of course activities is higher than its scheduled days number, 10 points are deducted for each shortcoming day. If a course has one or more activities split up in sessions which are scheduled on different days, points are attributed relative to the fraction of students for whom the spread is (sub)optimal. From this objective function, every existable valid timetable has an objective value within the upper and lower bounds of 1580 and -6001.

3 THREE ALGORITHMS

All three algorithms start off with randomized initial timetables and repeatedly applying swap-mutations, exchanging the contents of two randomly selected room slots, similar to a 2-opt in the TSP.

The HillClimber performs a swap-mutation each iteration, which is reverted only if it lowers the objective value of the new timetable.

PPA-1 keeps a population of 40 individuals in descending order of fitness. Every iteration, each individual produces offspring: 10, 5, 3 and 2 new individuals for its top 10% in the population, all mutated with a single swap-mutation, and 1 new individual for the remaining 90% which is subjected to three successive swap-mutations. From each individual and its *own* offspring (its 'family') the fittest individual remains in the population; the others are discarded. Thereby, PPA-1 is an almost direct translation from PPA for the TSP [8].

For the PPA-2 algorithm, there is no 10% - 90% division, but the number of offspring n_i and their mutability d_i is calculated from the smoother 'normalized fitness' value N_i from an individual's objective value $f(x_i)$ as

$$N_i = \frac{1}{2}(tanh(4 \cdot (\frac{f(x_{max}) - f(x_i)}{f(x_{max}) - f(x_{min})}) - 2) + 1) \tag{1}$$

in which $f(x_{max})$ and $f(x_{min})$ are the highest and lowest objective values in the population. The number of offspring for an individual is $n_i = \lceil n_{max} N_i r \rceil$, which are all mutated as $d_i = \lceil s_{max} \cdot r \cdot (1 - N_i) \rceil$ swap-mutations, in which n_{max} denotes the maximum number of swaps per offspring, s_{max} is the maximum allowable number of swap-mutations, and r is a random number in [0,1] which is redrawn every time it is invoked (anywhere). In this experiment, parameters $n_{max} = 10$ and $s_{max} = 20$ were used for all runs of PPA-2. Finally, all the newly generated offspring are added to the population, which is then sorted and retains only the best 40 individuals. Thereby, all three algorithms adopt an *elitist* approach, meaning the best objective value never decreases during a run (figure 1).

4 RESULTS, CONCLUSION & DISCUSSION

In this preliminary investigation, PPA-1 finds better timetables than the HillClimber, with maximum, average and minimum objective values of 1250, 1231 and 1209, over 1239, 1223 and 1201 after 10 runs of 200,000 function evaluations. PPA-2 performs best (1263, 1246, 1235), surprisingly outperforming both other algorithms after as many as 100,000 function evaluations. The absolute differences however are small and the rapid convergence of the HillClimber, traditionally susceptible to local maxima, could indicate that the objective function from this problem instance has a high degree of convexity. Contrarily, the fact that both PPA-algorithms – with their high-mutability offspring – persistently surpass the HillClimber after a very long time might indicate that local maxima are few and far between, which in terms might be due to the sheer vastness of this problem's state space [11].

In short, both PPA-algorithms seem to be plausible candidates for NP-hard optimization problem instances other than the TSP, such as this instance of the UCTP. Nonetheless, the gargantuan computational effort required for a better-than-HillClimbing solution raises some serious questions about the optimal parameterization of the proposed implementations. Furthermore, a more detailed state space survey including convexity measures, saddle-node maxima detection or symmetry-breaking might seriously improve final solutions, speed of convergence and total runtime.

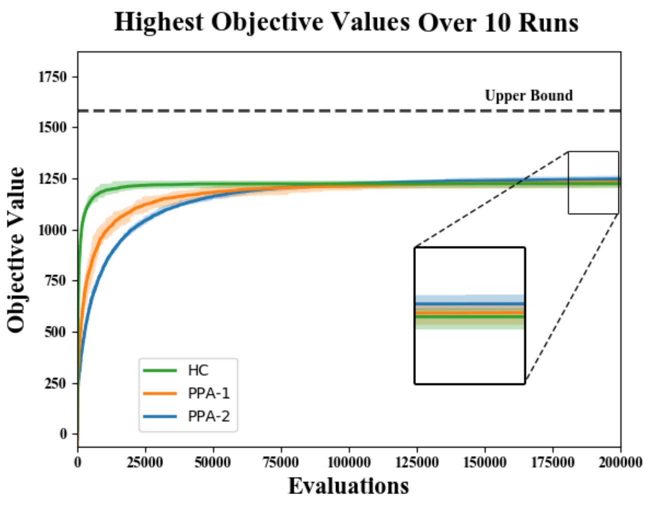

Figure 1: Though the HillClimber rapidly finds good solutions, PPA-1 and PPA-2 produce better timetables, but only after about 100,000 function evaluations. Solid lines are averages, transparent areas show the min/max objective values.

REFERENCES

[1] Pupong Pongcharoen, Weena Promtet, Pisal Yenradee, and Christian Hicks. Stochastic optimisation timetabling tool for university course scheduling. *International Journal of Production Economics*, 112(2):903–918, 2008.
[2] Hamed Babaei, Jaber Karimpour, and Amin Hadidi. A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86:43–59, 2015.
[3] Edmund K Burke, Graham Kendall, Mustafa Mısır, and Ender Özcan. Monte carlo hyper-heuristics for examination timetabling. *Annals of Operations Research*, 196(1):73–90, 2012.
[4] Olivia Rossi-Doria, Michael Sampels, Mauro Birattari, Marco Chiarandini, Marco Dorigo, Luca M Gambardella, Joshua Knowles, Max Manfrin, Monaldo Mastrolilli, Ben Paechter, et al. A comparison of the performance of different metaheuristics on the timetabling problem. In *International Conference on the Practice and Theory of Automated Timetabling*, pages 329–351. Springer, 2002.
[5] Krzysztof Socha, Michael Sampels, and Max Manfrin. Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. In *Workshops on Applications of Evolutionary Computation*, pages 334–345. Springer, 2003.
[6] Chong Keat Teoh, Antoni Wibowo, and Mohd Salihin Ngadiman. Review of state of the art for metaheuristic techniques in academic scheduling problems. *Artificial Intelligence Review*, 44(1):1–21, 2015.
[7] Meryem Cheraitia, Salim Haddadi, and Abdellah Salhi. Hybridizing plant propagation and local search for uncapacitated exam scheduling problems. *International Journal of of Services and Operations Management.*, 2017.
[8] Birsen İ Selamoğlu and Abdellah Salhi. The plant propagation algorithm for discrete optimisation: The case of the travelling salesman problem. In *Nature-inspired computation in engineering*, pages 43–61. Springer, 2016.
[9] Abdellah Salhi and Eric S Fraga. Nature-inspired optimisation approaches and the new plant propagation algorithm. 2011.
[10] Muhammad Sulaiman, Abdellah Salhi, Birsen Irem Selamoglu, and Omar Bahaaldin Kirikchi. A plant propagation algorithm for constrained engineering optimisation problems. *Mathematical Problems in Engineering*, 2014, 2014.
[11] Misha Paauw and Daan Van den Berg. Paintings, polygons and plant propagation. Springer, 2019.